



How To Analyze sPHENIX Geant4 Simulation Data

Liang Xue
Georgia State University



Outline

- Chris has already separated sPHENIX software from PHENIX software Universe. A lot of meaningful discussion have been made yesterday and today for the new sPHENIX framework.
- We have three different ways to analyze sPHENIX GEANT4 data. Here I will show you how I analyze sPHENIX data use my analysis modules.
 - g4simulation/g4eval
 - g4simulation/g4histos
 - **g4simulation/g4picoDst**

g4picoDst/G4SnglHit.cc ;	g4picoDst/G4SnglHit.h
g4picoDst/G4SnglTower.cc;	g4picoDst/G4SnglTower.h;
g4picoDst/G4SnglCluster.cc;	g4picoDst/G4SnglCluster.h
g4picoDst/G4SnglHTCContainer.cc;	g4picoDst/G4SnglHTCContainer.h
g4picoDst/mFillG4SnglHTCContainer.cc	g4picoDst/mFillG4SnglHTCContainer.h

<https://github.com/sPHENIX-Collaboration/coresoftware/tree/master/simulation/g4simulation>

How the module works



Read G4 files, need nodes:
HCALIN, ABSORBER_HCALIN,
HCALOUT, ABSORBER_HCALOUT

mFillG4SnglHTCContainer

G4SnglHTCContainer

G4SnglHit
G4SnglTower
G4SnglCluster

- Create node: G4SnglHTCContainer
- Fill single hit, tower, cluster information using functions:
`process_hit()`
`process_twr()`
`process_clr()`
- Incoming particle: PID, Energy, Phi, Theta, Px, Py, Pz
- `TClonesArray *G4SnglHits;`
`TClonesArray *G4SnglTowers;`
`TClonesArray *G4SnglClusters;`
- Single hit
- Single tower
- Single cluster



Write out root files which contains incoming particle info, single hit, tower, cluster information for further analysis.

How to setup the analysis macro

After the towering, and clustering modules, call the analysis modules as follows:

```
gSystem->Load("libg4picoDst.so");

mFillG4SnglHTCContainer* msngl = new mFillG4SnglHTCContainer();
msngl->CreateNode("G4SnglHTCContainer");
msngl->Set_MakeHit(false);
msngl->Set_MakeTower(true);
msngl->Set_MakeCluster(true);
msngl->AddNode("CEMC",0);
msngl->AddNode("ABSORBER_CEMC",1);
msngl->AddNode("HCALIN",2);
msngl->AddNode("ABSORBER_HCALIN",3);
msngl->AddNode("HCALOUT",4);
msngl->AddNode("ABSORBER_HCALOUT",5);
se->registerSubsystem(msngl);

Fun4AllDstOutputManager *out = new Fun4AllDstOutputManager("DSTOUT", outputFile);
out->AddNode("G4SnglHTCContainer");
se->registerOutputManager(out);
```

Turn on/off hit, tower, cluster output

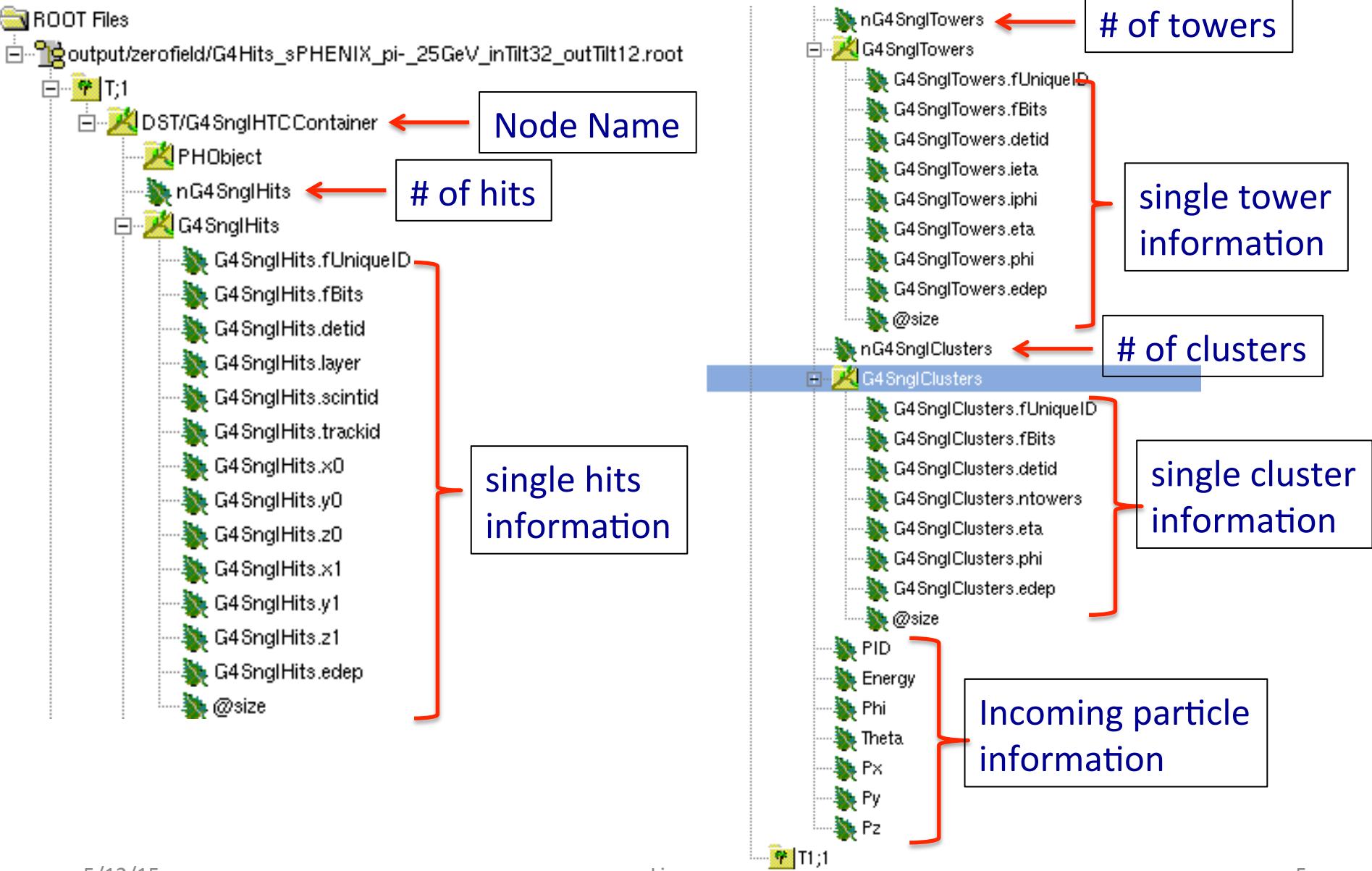
Add required nodes

Register output

An Example:

[https://www.phenix.bnl.gov/viewvc/viewvc.cgi/phenix/simulation/g4simulation/macros/
Fun4All_G4_sPHENIX_Xue.C?view=markup](https://www.phenix.bnl.gov/viewvc/viewvc.cgi/phenix/simulation/g4simulation/macros/Fun4All_G4_sPHENIX_Xue.C?view=markup)

ROOT Output Format



Further analysis

Define variables:

```
Float_t Energy;
Float_t Phi;
Float_t Theta;
Float_t Px;
Float_t Py;
Float_t Pz;
UInt_t nG4SnglTowers;
Int_t G4SnglTowers_;
UInt_t nG4SnglClusters;
Int_t G4SnglClusters_;
```



```
Int_t twr_detid[MAXTWRS];
Float_t twr_ieta[MAXTWRS];
Float_t twr_iphi[MAXTWRS];
Float_t twr_eta[MAXTWRS];
Float_t twr_phi[MAXTWRS];
Float_t twr_edep[MAXTWRS];
```



```
Int_t clr_detid[MAXCLRS];
Int_t clr_ntowers[MAXCLRS];
Float_t clr_eta[MAXCLRS];
Float_t clr_phi[MAXCLRS];
Float_t clr_edep[MAXCLRS];
```

Set branch address:

```
chain->SetMakeClass(1);
chain->SetBranchStatus("*",0); // disable all branches
chain->SetBranchAddress("Energy", &Energy);
chain->SetBranchAddress("Phi", &Phi);
chain->SetBranchAddress("Theta", &Theta);
chain->SetBranchAddress("Px", &Px);
chain->SetBranchAddress("Py", &Py);
chain->SetBranchAddress("Pz", &Pz);
chain->SetBranchAddress("nG4SnglTowers", &nG4SnglTowers);
chain->SetBranchAddress("G4SnglTowers", &G4SnglTowers_);
chain->SetBranchAddress("nG4SnglClusters", &nG4SnglClusters);
chain->SetBranchAddress("G4SnglClusters", &G4SnglClusters_);

chain->SetBranchAddress("G4SnglTowers.detid", twr_detid);
chain->SetBranchAddress("G4SnglTowers.ieta", twr_ieta);
chain->SetBranchAddress("G4SnglTowers.iphi", twr_iphi);
chain->SetBranchAddress("G4SnglTowers.eta", twr_eta);
chain->SetBranchAddress("G4SnglTowers.phi", twr_phi);
chain->SetBranchAddress("G4SnglTowers.edep", twr_edep);

chain->SetBranchAddress("G4SnglClusters.detid", clr_detid);
chain->SetBranchAddress("G4SnglClusters.ntowers", clr_ntowers);
chain->SetBranchAddress("G4SnglClusters.eta", clr_eta);
chain->SetBranchAddress("G4SnglClusters.phi", clr_phi);
chain->SetBranchAddress("G4SnglClusters.edep", clr_edep);
```

An example towards results

```
for (int i=0; i<mNEvts; i++) {  
    chain->GetEntry(i);
```

Event loop, PID, energy, momentum

```
double ClrE01(0.), ClrE02(0.), ClrE03(0.);
```

```
for (int j=0; j<nG4SnglClusters; j++){
```

```
    if(clr_detid[j]==0){
```

```
        hClrEEMCal->Fill(clr_edep[j]);
```

```
        hClrEMCalNTwr->Fill(clr_ntowers[j]);
```

```
        ClrE01 += clr_edep[j];
```

```
}
```

```
else if(clr_detid[j]==2){
```

```
    hClrEHCAL1->Fill(clr_edep[j]);
```

```
    hClrHCal1NTwr->Fill(clr_ntowers[j]);
```

```
    ClrE02 += clr_edep[j];
```

```
}
```

```
else if(clr_detid[j]==4){
```

```
    hClrEHCAL2->Fill(clr_edep[j]);
```

```
    hClrHCal2NTwr->Fill(clr_ntowers[j]);
```

```
    ClrE03 += clr_edep[j];
```

```
}
```

```
}
```

```
....
```

Cluster loops

Detid==0 is EMCAL

Detid==2 is INNER HCAL

Detid==2 is OUTER HCAL

Scaled with sampling fraction factors, and then sum over reconstructed energy from EMCAL, H1, and H2, evaluate performance.

Summary

- We have summarized how to analyze sPHENIX GEANT4 data as an advanced users.
- The modules talked about in this presentation has been used for EMCAL, HCal, and Hcal prototype GEANT4 simulation data analysis.
- Look forward to see you again at **Atlanta HCal workshop: 05/18/2015 – 05/21/2015.**